# ! pip install kaggle

# ! mkdir ~/.kaggle

# ! cp kaggle.json ~/.kaggle/

# ! chmod 600 ~/.kaggle/kaggle.json

# ! kaggle competitions download -c rsna-pneumonia-detection-challenge

# ! unzip rsna-pneumonia-detection-challenge.zip

train\_labels = pd.read\_csv('stage\_2\_train\_labels.csv')

classInfo = pd.read\_csv('stage\_2\_detailed\_class\_info.csv')

merged = pd.merge(left = classInfo, right = train\_labels, how = 'left', on = 'patientId') # Merge

merged = merged.drop\_duplicates() # Remove duplicates

merged.head()

merged = merged.fillna(0)

merged.head()

def get\_names(path):

names = []

for root, dirnames, filenames in os.walk(path):

for filename in filenames:

\_, ext = os.path.splitext(filename)

if ext in ['.dcm']:

names.append(filename)

return names

def convert\_dcm\_jpg(name):

im = pydicom.dcmread('stage\_2\_train\_images/'+name)

im = im.pixel\_array.astype(float)

rescaled\_image = (np.maximum(im,0)/im.max())\*255 # float pixels

final\_image = np.uint8(rescaled\_image) # integers pixels

final\_image = Image.fromarray(final\_image)

return final\_image

# names = get\_names('stage\_2\_train\_images/')

# names

# for name in names:

# image = convert\_dcm\_jpg(name)

# image.save('/content/drive/MyDrive/Colab Notebooks/converted\_train\_images/'+name+'.jpg')

# images = os.listdir('/content/drive/MyDrive/Colab Notebooks/converted\_train\_images')

# images[0]

X=[]

Y=[]

W=[]

H=[]

for i in range(len(train\_labels)):

x1=Merged['x'][i:i+1]

y1=Merged['y'][i:i+1]

w1=Merged['width'][i:i+1]

h1=Merged['height'][i:i+1]

X.append(x1)

Y.append(y1)

W.append(w1)

H.append(h1)

X = [int(i) for i in X]

Y = [int(i) for i in Y]

W = [int(i) for i in W]

H = [int(i) for i in H]

for i in range(merged.shape[0]):

merged.at[ i , 'mask'] = coordinates[i]

from tqdm.notebook import tqdm

for idx,row in tqdm(merged.iterrows(),total=len(merged)):

ls = f"{row['x']} {row['y']} {row['width']} {row['height']}"

merged.at[ idx , 'MASK'] = ls

# merged.reset\_index(inplace=True)

# for i in range(Merged.shape[0]):

# id = Merged['patientId'][i]

# path = '/content/drive/MyDrive/Colab Notebooks/converted\_train\_images/'+id+'.dcm.jpg'

# Merged.at[i,'path']=path

batch\_size=1

IMG\_SIZE=512

train\_gen = DataGenerator(list(Merged.patientId),

list(Merged.MASK),

batch\_size=batch\_size,

IMG\_SIZE=IMG\_SIZE,

shuffle=True)

img = train\_gen[0][0][0]

# plt.imshow(img/img.max())

# img.astype('float')/img.max()

img.min()

plt.imshow(train\_gen[1][0][2][:,:,-1],cmap='gray')

plt.imshow(train\_gen[1][1][2][:,:,-1],cmap='jet',alpha=0.3)

X\_train, y\_train, X\_valid, y\_valid, X\_test, y\_test = train\_valid\_test\_split(Merged, target = 'Target', train\_size=0.7, valid\_size=0.15, test\_size=0.15)

X\_train.reset\_index(drop =True,inplace=True)

y\_train.reset\_index(drop =True,inplace=True)

X\_valid.reset\_index(drop =True,inplace=True)

y\_valid.reset\_index(drop =True,inplace=True)

X\_test.reset\_index(drop =True,inplace=True)

y\_test.reset\_index(drop =True,inplace=True)

X\_train.to\_csv('/content/drive/MyDrive/Data/X\_train.csv')

y\_train.to\_csv('/content/drive/MyDrive/Data/y\_train.csv')

X\_valid.to\_csv('/content/drive/MyDrive/Data/X\_valid.csv')

y\_valid.to\_csv('/content/drive/MyDrive/Data/y\_valid.csv')

X\_test.to\_csv('/content/drive/MyDrive/Data/X\_test.csv')

y\_test.to\_csv('/content/drive/MyDrive/Data/y\_test.csv')

img = train\_gen[0][0][0]

plt.imshow(img/img.max())

img.astype('float')/img.max()

newdf = pd.DataFrame(columns=['patientId','masks'])

unq\_IDs = Merged.patientId.unique()

newdf['patientId'] = unq\_IDs

def multiple\_masks(ID):

x=[]

y=[]

w=[]

h=[]

li=[]

for i in range(Merged.shape[0]):

if(Merged['patientId'][i] == ID):

x.append(Merged['x'][i])

y.append(Merged['y'][i])

w.append(Merged['width'][i])

h.append(Merged['height'][i])

li.append(x)

li.append(y)

li.append(w)

li.append(h)

return li

from collections import defaultdict

Masks = defaultdict(list)

for i in tqdm(range(newdf.shape[0])):

li = multiple\_masks(newdf['patientId'][i])

Masks['x'].append(li[0])

Masks['y'].append(li[1])

Masks['w'].append(li[2])

Masks['h'].append(li[3])

print(len(Masks))

newdf['x'] = Masks['x']

newdf['y'] = Masks['y']

newdf['w'] = Masks['w']

newdf['h'] = Masks['h']

masks = []

for i in tqdm(range(newdf.shape[0])):

fli=[]

tli=[]

for j in range(len(eval(newdf['x'].iloc[i]))):

tli.append(eval(newdf['x'].iloc[i])[j])

tli.append(eval(newdf['y'].iloc[i])[j])

tli.append(eval(newdf['w'].iloc[i])[j])

tli.append(eval(newdf['h'].iloc[i])[j])

fli.append(tli)

tli=[]

masks.append(fli)

newdf['Masks'] = masks